MINISTRY OF HIGHER EDUCATION

RANA UNIVERSITY

FACULTY OF COMPUTER SCIENCE

IT DEPARTMENT

Linux Administration

7th Semester

Prepared By: Baitullah Malakhil

2025

Course Overview

Linux has become the backbone of modern computing, powering everything from mobile devices to the world's largest data centers. As organizations increasingly rely on open-source technologies for performance, security, and cost-effectiveness, the demand for skilled Linux administrators continues to grow. This course has been designed to equip learners with the practical knowledge and technical expertise required to install, configure, and manage servers on **Red Hat Enterprise Linux (RHEL 7)**.

The course takes a **step-by-step approach**, beginning with the fundamentals of Linux administration and gradually progressing towards advanced network service configurations. Unlike purely theoretical modules, this handout emphasizes **hands-on practice**, ensuring that students not only learn commands but also understand their practical applications in real-world environments.

The curriculum introduces learners to essential system management concepts, such as **user management, file permissions, and process handling**, before advancing to the configuration of **network services including DNS, DHCP, FTP, Samba, NTP, and web servers**. Special focus is given to tools like **Vim**, service management with **systemctl**, and secure communication using **firewalls and SELinux policies**. Each chapter includes **real command-line examples**, preparing learners for tasks they are likely to face in professional IT environments.

Course Philosophy

The course is structured around three key principles:

1. **Practical Relevance** – Every topic is presented with real-life use cases. For example, DHCP configuration is not just explained but also demonstrated step by step for both server and client systems.
2. **Progressive Learning** – Learners start with simpler concepts such as text editors and file systems and then build up to more advanced topics like server deployment and network troubleshooting.
3. **Hands-on Experience** – Since Linux is best learned by doing, each chapter provides direct commands and configuration tasks that students can replicate on their own systems.

Skills You Will Gain

By the end of this course, learners will be able to:

- Understand and navigate the Linux operating system with confidence.
- Use and configure different editors, including **Vim and Gedit**, for text and code editing.
- Manage services, processes, and system logs effectively.
- Configure and secure a variety of **network services** (DNS, DHCP, NTP, FTP, Samba, etc.).
- Set up and manage **firewalls** using firewalld and ensure secure access.

- Apply best practices in **system administration and troubleshooting**.
- Build strong foundations for pursuing advanced certifications like **RHCSA (Red Hat Certified System Administrator)**.

Who Should Take This Course?

This course is particularly suited for:

- **Undergraduate and graduate students** of Computer Science and IT wish to strengthen their Linux knowledge.
- **System administrators** are looking to expand their skills in server configuration and management.
- **Networking professionals** interested in deploying Linux-based network services.
- **Beginner to intermediate learners** who want structured, practical exposure to RHEL server administration.

Table of Contents

Introduction to the course 10
1. Chapter I — Concept of Open Source 11
1.1. Open-Source Philosophy 11
1.2. Benefits of Open Source 11
1.3. Linux vs Windows 11
1.4. Main Linux Distributions 11
1.5. Linux Applications 14
1.6. Open-Source Licensing 14
1.7. Freedom and Business 14
1.8. Bash Shell: 14
1.9. RedHat Certificates: 16
1.10. RHEL Installation on VMware (Workstation / Player) 16
1.11. Accessing the Terminal & RHEL 7 Terminals 17
1.12. Some Basic Commands 18
1.13. Practice Exercises 19
2. Chapter II — Basic Commands 20
2.1. Linux Runlevels (SysV init) 20
2.2. RHEL Boot Process 20
2.3. Some basic Linux commands 22
3. Chapter III — File System Hierarchy & Navigation 26
3.1. File System Hierarchy 26
3.2. Changing Directories 26
3.3. Absolute Path 27
3.4. Relative Path 27
3.5. ✅ Quick Recap 27
4. Chapter IV — Managing Files & Directories 28
4.1. Copying the contents of one file into another 28
4.2. Copying one or more files from one location into another 28
4.3. Copying one or more directories from one location into another 28
4.4. Moving the text of one file into another (rename file) 28
4.5. Moving one or more files from one location to another 29
4.6. Moving one or more directories from one location into another 29
4.7. Deleting one or more files 29

4.8. Deleting one or more files with -f (force option) 29
4.9. 9. Deleting one or more directories along with the contents 29
4.10. Deleting empty directories 30
4.11. 📌 Quick Summary Table 30
5. Chapter V — Getting Help in Linux 31
5.1. Tab Completion 31
5.2. Double Tab (Tab Tab) 31
5.3. man – Manual Pages 31
5.4. info 32
5.5. pinfo 32
5.6. whatis 32
5.7. Linux Package Documentation Path 32
5.8. Getting Help from Red Hat Support 32
5.9. find 33
5.10. locate 33
5.11. Generating SoS Report 33
5.12. --help Option 33
5.13. whoami 33
5.14. whereis 34
5.15. pwd – Print Working Directory 34
5.16. 📌 Quick Reference Table 34
5.17. ⚡ Tip for Students: 35
6. Chapter VI — vim Editor 36
6.1. What is an Editor? 36
6.2. Some Common Editors in Linux 36
6.3. Vim Editor 36
6.4. Vim Modes 36
6.5. Switching Modes 36
6.6. Copy, Cut, Paste 37
6.7. Gedit Editor 37
6.8. Using Gedit 37
6.9. 📌 Quick Recap 38
6.10. ⚡ Tip for Students: 38
7. Chapter VII — Local User & Group Account Management 39

7.1. UID and GID 39
7.2. Important System Files 39
7.3. Creating Users 40
7.4. 5. Deleting Users 40
7.5. 6. Creating Groups 40
7.6. 7. Adding Users to Groups 40
7.7. 8. Giving User Privilege to Perform Root Tasks 41
7.8. 9. Password Policies using chage 41
7.9. 10. Messaging Commands: wall and write 41
7.10. 📌 Quick Reference Table 41
7.11. ⚡ Tips for Students: 42
8. Chapter VIII — File and Directory Permissions 43
8.1. Default File and Directory Permissions 43
8.2. Permission Representation and Calculation 43
8.3. Umask Value 44
8.4. Changing Permissions with chmod 44
8.5. Changing Ownership 45
8.6. 📌 Quick Reference Table 45
8.7. ⚡ Tips for Students: 45
9. Chapter IX — Linux Processes 46
9.1. Viewing Processes 46
9.2. top – Real-time Process Monitor 46
9.3. tty – Current Terminal 47
9.4. Killing Processes 47
9.5. Summary Table of Commands 48
9.6. ⚡ Tips for Students: 48
10. Chapter X — Linux Services and Daemons 49
10.1. Location of Services 49
10.2. Listing Services and Daemons 49
10.3. Managing Services in RHEL7 50
10.4. RHEL6 (Legacy commands) 50
10.5. Disabled vs Stopped Services 50
10.6. Using Mask Command 50
10.7. Quick Recap Table (RHEL7) 51

10.8. ⚡ Tips for Students: 51
11. Chapter XI — Linux Logs 52
11.1. Why logs are important: 52
11.2. Log File Locations 52
11.3. Viewing Logs Using cat 52
11.4. Viewing Logs Using journalctl 52
11.5. Date and Time Related Commands 53
11.6. Tips for Students 53
11.7. Quick Reference Table 53
11.8. ⚡ Tip for Students: 54
12. Chapter XII — Yum Server 55
12.1. Step 1: Copy RHEL DVD to Local Directory 55
12.2. Step 2: Mount the ISO 55
12.3. Step 3: Configure Local YUM Repository 55
12.4. Step 4: Disable RHN Plugin (Optional) 56
12.5. Step 5: Clean and Update YUM 56
12.6. Step 6: Installing Packages Using YUM 56
12.7. Step 7: Managing Installed Packages 56
12.8. Step 8: Updating Kernel and System 57
12.9. Step 9: Enabling/Disabling Local Repo and RHN Plugin 57
12.10. Tips for Students 57
13. Chapter XIII — Basic Networking 58
13.1. Network Configuration Files 58
13.2. Hostname Management 58
13.3. Viewing Network Information 58
13.4. Managing Network Connections (nmcli) 59
13.5. Testing Network Connectivity 59
13.6. Firewall Management 59
13.7. Quick Tips for Students 60
13.8. Summary Table of Useful Commands 60
14. Chapter XIV — Linux Firewall 61
14.1. Check Firewall Installation and Status 61
14.2. Zones and Services 61
14.3. Adding/Removing Ports and Services 62
14.4. Forwarding Ports 62

14.5. Rich Rules 62
14.6. Panic Mode 63
14.7. Assigning Interfaces to Zones 63
14.8. Default Zone Management 63
14.9. Firewalld vs Iptables 63
14.10. GUI Firewall Management 63
14.11. Quick Reference Table 63
14.12. Tip for Students: 64
15. Chapter XV — DNS Server 65
15.1. Network Preparation 65
15.2. Install and Verify DNS Packages 65
15.3. Firewall Configuration for DNS 65
15.4. Configure DNS Service (named) 66
15.5. Forward and Reverse Zone Files 66
15.6. Validate and Restart DNS 67
15.7. Test DNS Resolution 67
15.8. Configure Resolver on Clients 67
15.9. Summary 67
15.10. Tip for Students: 68
16. Chapter XVI — DHCP Server 69
16.1. Network Preparation 69
16.2. Install DHCP Packages 69
16.3. Firewall Configuration 69
16.4. Configure DHCP Server 70
16.5. Start and Manage DHCP Service 70
16.6. Configure Clients for DHCP 70
16.7. Verify DHCP Operation 71
16.8. Summary 71
17. Chapter XVII — Web Server 72
18. Chapter XVIII — Web Server 75
18.1. Network Preparation 75
18.2. Install Samba Server 75
18.3. Configure Shared Directory 76
18.4. Configure Samba 76
18.5. Accessing Samba Share from RHEL Client 76

18.6. Accessing Samba Share from Windows Client 77
18.7. Summary 77
19. Chapter XIX — vsFTP Server 78
19.1. Network Preparation 78
19.2. Install vsFTP Server 78
19.3. Configure FTP Directory 79
19.4. Configure vsFTP Server 79
19.5. Accessing FTP Server from Linux Client 79
19.6. Accessing FTP Server from Windows Client 80
19.7. Non-Anonymous FTP Access 80
19.8. Secure File Copy between Linux Systems 80
19.9. Summary 80
20. Chapter XX — NTP Server 81
20.1. Network Preparation 81
20.2. Install NTP Packages 81
20.3. Configure Firewall for NTP 81
20.4. Configure NTP Server 82
20.5. Configure NTP on Linux Client 82
20.6. Configure NTP on Windows Client 83
20.7. Summary 83
Conclusion 84
References 85

Introduction to the course

This course provides a comprehensive, hands-on introduction to **Linux system administration and network services**. It is designed for students and IT professionals who want to gain practical knowledge of configuring, managing, and securing Linux servers.

Throughout this course, learners will explore key aspects of **RHEL 7**, including system services, networking, security, and server deployment, with a strong emphasis on real-world applications.

Learning Outcomes

By the end of this course, learners will be able to:

- Understand the **Linux operating system structure**, services, and daemons.
- Manage and monitor **Linux logs** for security and troubleshooting.
- Configure and maintain a **YUM repository** for package management.
- Set up and manage **basic networking** on RHEL 7 servers.
- Implement and manage **firewalls using firewalld and iptables**.
- Deploy and configure **DNS, DHCP, and NTP servers**.
- Install and configure **web servers** (Apache) for hosting websites.
- Implement **file sharing using Samba** for Linux and Windows clients.
- Set up **FTP servers** (vsFTPd) for secure file transfer.
- Apply practical **firewall rules, port management, and rich rules** for enhanced network security.
- Troubleshoot and verify configurations for network services and server applications.

This course combines theory with **step-by-step practical commands**, ensuring learners gain both understanding and hands-on skills.

1. Chapter I — Concept of Open Source

1.1. Open-Source Philosophy

Open-source software refers to programs whose source code is made freely available for anyone to inspect, modify, enhance, and redistribute. The philosophy emphasizes freedom and community-driven development rather than closed, proprietary control. It is not just about cost, but about rights and transparency.

Key aspects of open source software include:
- Freedom to use: Anyone can run the software for any purpose.
- Freedom to study: The source code is open for analysis and learning.
- Freedom to modify: Developers can adapt the software to their needs.
- Freedom to share: Copies can be freely distributed to others.

Analogy: Getting free cookies is like freeware (no cost, but no recipe). Getting free cookies with the recipe is like open source (you can study, modify, and bake your own versions).

1.2. Benefits of Open Source

- Cost-effective: Typically, free to use, reducing licensing expenses.
- Transparency: Source code is auditable for security and reliability.
- Community support: Developers and users worldwide contribute patches, improvements, and help.
- Flexibility: Can be customized for individual or organizational needs.
- Innovation: Rapid updates and improvements driven by a large community.

1.3. Linux vs Windows

Linux	Windows
Open source; free distributions like Ubuntu, CentOS	Closed source; requires paid licenses
Highly customizable; users can modify kernel and applications	Limited customization; proprietary core
Stable and secure; widely used in servers	User-friendly; widely used in desktops
Strong community support and forums	Official vendor support from Microsoft
Command-line focused, though GUIs exist	GUI-focused, though has PowerShell CLI
Runs efficiently on older hardware	Generally, it requires newer hardware
Free updates provided by community	Updates provided by Microsoft at scheduled intervals

1.4. Main Linux Distributions

🖥️ **1. Client (Desktop) Distributions**

Purpose: Designed for daily use by individuals—user-friendly interfaces, multimedia support, office tools, etc.

Key Features:

- Graphical User Interface (GUI) by default.
- Pre-installed software like browsers, document editors, media players.
- Focus on ease of use, driver support, and hardware compatibility.

Popular Client Distros:

Distribution	Description
Ubuntu Desktop	Beginner-friendly, wide community support, used in education and general computing.
Linux Mint	Based on Ubuntu/Debian, lightweight and familiar interface for Windows users.
Fedora Workstation	Cutting-edge software, GNOME desktop, upstream for Red Hat technologies.
elementary OS	Mac-like interface, design-focused, good for casual users.

🖧 2. Server Distributions

Purpose: Designed for stability, performance, and security in server environments like web hosting, databases, cloud infrastructure, etc.

Key Features:

- Minimal GUI (or none at all).
- Long-term support (LTS) cycles.
- Package management geared toward performance and up time.
- Optimized for headless (remote) administration.

Popular Server Distros:

Distribution	Description
Red Hat Enterprise Linux (RHEL)	Enterprise-grade, commercial support, widely used in businesses.
CentOS Stream / AlmaLinux / Rocky Linux	Community-supported RHEL alternatives.
Ubuntu Server	Free and popular in cloud environments.
Debian	Extremely stable, often used as the base for other distros.
SUSE Linux Enterprise Server (SLES)	Commercial, used in corporate environments and SAP systems.

📝 **Note**: Many client and server distros share the same core but differ in packages and configurations.

🛡️ 3. Security / Penetration Testing Distributions

Purpose: Specialized distributions for cybersecurity professionals, ethical hackers, and security researchers.

Key Features:

- Bundled with security and forensic tools.
- Live boot options for forensics.
- Frequently updated with new tools.

Popular Security Distros:

Distribution	Description
Kali Linux	Maintained by Offensive Security; it includes hundreds of penetration testing tools.
Parrot Security OS	Lighter than Kali; supports forensics, development, and security testing.
BackBox	Ubuntu-based, simpler security testing environment.
Tails	Privacy-focused; routes all traffic through Tor; designed for anonymity.

Summary Table

Type	Goal	Example Distros
Client	User-friendly, GUI-based	Ubuntu Desktop, Linux Mint, Fedora
Server	Performance, uptime	RHEL, Ubuntu Server, Debian Server
Security	Ethical hacking, forensics	Kali Linux, Parrot OS, Tails

A **Linux distribution (distro)** is a packaged version of the Linux operating system. Each distribution includes the Linux kernel, GNU tools, system libraries, and additional software.

- **RHEL (Red Hat Enterprise Linux):** Commercial, enterprise-grade with professional support.
- **CentOS:** Free, community-supported rebuilding of RHEL.
- **Ubuntu:** Popular desktop and server distribution, user-friendly.
- **Debian:** Stable, community-focused, widely used as a base for other distros.
- **SUSE Linux Enterprise:** Widely used in Europe, enterprise-level support.

Linux also exists in embedded devices such as routers, Android phones, smart TVs, and car systems.

1.5. Linux Applications

Linux supports a wide variety of applications:

- **Desktop Applications:** LibreOffice, OpenOffice, Firefox, Thunderbird, Chromium.
- **Server Applications:** Apache (web server), Samba (file sharing with Windows), NFS (Linux/Unix file sharing), MySQL (database), Postfix (mail server).
- **Programming Tools:** C, C++, Java, Python, PHP, Perl, and shell scripting (Bash).
- **System Administration Tools:** yum (package manager), webmin (web-based administration), and GUI tools.

Example:
A Linux server can host a website using **Apache**, manage databases with **MySQL**, and be administered remotely using **SSH**.

1.6. Open-Source Licensing

Licenses define the rights and restrictions of using open-source software.

- **Free Software Foundation (FSF):** Founded by Richard Stallman, advocates software freedom.
    - Provides licenses like **GPL** (General Public License) and **LGPL** (Lesser General Public License).
    - Four essential freedoms: Run, study, share, and modify software.
- **Open-Source Initiative (OSI):** Defines what qualifies as open source.
    - Recognizes licenses such as GPL, MIT, BSD, and Apache.
    - Has 10 rules that licenses must follow to be considered open source.

1.7. Freedom and Business

Even though open source is free to use, companies can still earn revenue by:

- Selling **support and consulting services**.
- Offering **dual licensing** — one version open source, another commercial.
- Bundling open-source software with **hardware** (e.g., routers with Linux).

Example:
Red Hat sells RHEL subscriptions that include certified updates, training, and support — even though the source code is available for free.

1.8. Bash Shell:

Bash stands for **Bourne Again SHell**. It is a **command-line interpreter** or **shell** used to interact with the operating system.

It allows users to type commands, run scripts, manage files, automate tasks, and control system behavioral through text commands.

Bash is:

- A **free and open-source** shell.
- An **improved version** of the original Unix **Bourne shell (sh)**.
- Developed by the **GNU Project**.

🔧 Features of Bash

- Command history (↑ ↓ keys to scroll through previous commands)
- Tab completion (auto-completes filenames, commands)
- Shell scripting (automating tasks with .sh files)
- Variables and loops
- Built-in commands like echo, cd, pwd, ls, export

Operating Systems that Support Bash

Bash is the **default or available shell** on many Unix-like systems:

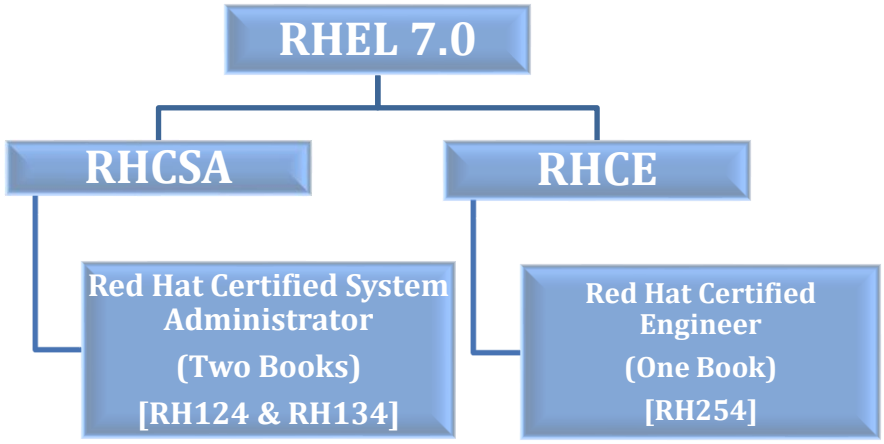
Operating System	Bash Support?	Notes
Linux (All distros)	✅ Yes	Bash is usually the default shell
macOS	✅ Yes (zsh by default now)	Older versions used Bash as default
Ubuntu	✅ Yes	Default shell is Bash
Debian	✅ Yes	Bash is default
Red Hat / RHEL	✅ Yes	Bash is standard
Fedora	✅ Yes	Bash is default
Alpine Linux	✅ Optional	Uses ash by default; Bash can be installed
Windows 10/11	✅ Yes (via WSL or Git Bash)	Not native, but fully supported through Windows Subsystem for Linux or tools like Git Bash
BSD systems (e.g., FreeBSD)	✅ Optional	Not default, but can be installed

- RHEL supports over **22 languages**.
- The default shell in RHEL 7 is **Bash (Bourne Again SHell)**, written by Brian Fox.
- Bash is also the default shell in macOS.

1.9. RedHat Certificates:

- **RHCSA (Red Hat Certified System Administrator)**
  - Entry-level Red Hat certification.
  - Validates skills in **Linux administration**: installation, configuration, managing users, files, storage, security, and basic troubleshooting.
  - Good for beginners or system admins starting with Red Hat Enterprise Linux.
- **RHCE (Red Hat Certified Engineer)**
  - Advanced certification, usually after RHCSA.
  - Focus on **automation and enterprise-level tasks**: networking services, security, managing systems at scale, automation with Ansible, and advanced troubleshooting.
  - More suitable for experienced Linux admins and engineers.

In short: **RHCSA = core Linux admin skills**, while **RHCE = advanced, automation, and enterprise-level skills**.



1.10. RHEL Installation on VMware (Workstation / Player)

---

1. Prerequisites

Before beginning the installation, make sure students have:

- A computer with at least 8GB RAM (16GB recommended).
- VMware Workstation / VMware Player installed.
- RHEL ISO file (e.g., rhel-9.x-x86_64-dvd.iso) – can be downloaded from the Red Hat Customer Portal (requires a free developer account).
- Basic knowledge of virtualization concepts.

---

2. Creating a New Virtual Machine

Step by step:

1. **Open VMware Workstation / Player.**
2. Click **"Create a New Virtual Machine"**.
3. Choose **"Typical (recommended)"**.
4. Select **"Installer disc image file (iso)"** and browse for your RHEL ISO.
5. Click **Next** and select **Linux** as guest OS, with **Red Hat Enterprise Linux 9 or 8** depending on ISO.
6. Name the VM, e.g., "RHEL9-Student".
7. Set maximum disk size (e.g., 40 GB) and select **"Store virtual disk as a single file"**.
8. Click **Customize Hardware** to:
    - Allocate at least **2GB RAM** (4GB+ recommended).
    - Set **2 processors**.
    - Ensure **network adapter** is set to NAT or Bridged (explain difference).
9. Click **Finish**.

---

3. Starting the Installation

1. Power on the VM.
2. Select **"Install Red Hat Enterprise Linux 9.x"**.
3. Wait for the graphical installer to load (Anaconda installer).

---

4. Installation Setup in Anaconda

Configuration areas to cover:

- **Language Selection:** Choose English (or as preferred).
- **Installation Destination:**
    - Choose the VM disk.
    - Let the installer auto-partition (explain this briefly).
- **Software Selection:**
    - Choose **Server with GUI** or **Workstation** (explain options).
- **Network & Hostname:**
    - Enable the network interface.
    - Set a hostname like rhel9.localdomain.
- **User Settings:**
    - Set **root password**.
    - Create a student user account (e.g., student).

Once done, click **"Begin Installation"** and wait.

1.11. Accessing the Terminal & RHEL 7 Terminals

The **terminal** (or command line interface) is where users interact with the system.

When opening a terminal, you may see a prompt like:

[root@localhost ~]#

- **root** → Current username
- **localhost** → Hostname of the system
- **~** → Home directory
- **#** → Root (administrator) user
- **$** → Normal user

RHEL provides **11 virtual terminals** (text-based), but only 6 are active by default.

- Multiple users can log in on different terminals simultaneously.
- There is **1 graphical terminal (GUI)**.

Shortcuts to switch terminals:

- Alt + Ctrl + F1 → Terminal 1
- Alt + Ctrl + F2 → Terminal 2
- … up to Alt + Ctrl + F6

Note:
In older RHEL versions, the GUI ran on terminal 7 (Ctrl+Alt+F7).

1.12. Some Basic Commands

Command	Description	Example
shutdown -h now	Shutdown system immediately	shutdown -h now
shutdown -r now	Restart system immediately	shutdown -r now
reboot	Reboot system	reboot
whoami	Show current logged-in user	whoami
who / w	Show users currently logged in	who / w
halt / poweroff / init 0	Alternative shutdown methods	halt / poweroff / init 0
init 6	Reboot system via runlevel command	init 6

```
#shutdown -h +3     # Shutdown after 3 minutes
#shutdown -r now    # Restart immediately
#shutdown -c        # Cancel shutdown
#reboot
#reboot -f          # Force reboot
#who
#w
#whoami
#halt
#poweroff
#systemctl poweroff
#init 0    # Shutdown
#init 6    # Reboot
```

1.13. Practice Exercises

1. Write a short note comparing Linux and Windows.
2. List three benefits of open-source software.
3. Practice switching between virtual consoles using Ctrl+Alt+F keys.
4. Use shutdown and reboot commands on a test system.
5. Explore the 'who' and 'w' commands to check logged-in users.

2. Chapter II — Basic Commands

2.1. Linux Runlevels (SysV init)

Runlevels are **modes of operation** that tell the system which services/processes to run after boot.

Standard Runlevels:

1. **Runlevel 0 – Halt**
   - Shuts down the system.
   - Entering runlevel 0 = power off.
2. **Runlevel 1 – Single-user mode**
   - Maintenance mode for the **root user only**.
   - No networking, no multi-user environment.
   - Used for troubleshooting (like fixing filesystem errors).
3. **Runlevel 2 – Multi-user, no networking**
   - Rarely used in most Linux distros.
   - Multiple users allowed, but networking is disabled.
4. **Runlevel 3 – Multi-user, text mode**
   - Full multi-user environment.
   - Networking enabled.
   - No graphical interface, only CLI (common for servers).
5. **Runlevel 4 – Unused/Custom**
   - Not defined by default.
   - Sysadmins can configure this for custom purposes.
6. **Runlevel 5 – Multi-user, GUI mode**
   - Same as runlevel 3 + graphical interface (X11, GNOME, KDE).
   - Default for desktops.
7. **Runlevel 6 – Reboot**
   - Reboots the system immediately.
   - ⚠️ Dangerous to set as default!

2.2. RHEL Boot Process

a. BIOS/UEFI (Firmware Stage)

- When you power on the system, **BIOS (older PCs)** or **UEFI (newer systems)** runs.
- Performs **POST (Power-On Self-Test)** → checks CPU, RAM, disks, etc.
- Then looks for a **bootable device** (HDD, SSD, USB, CD, network).
- Hands control to the **bootloader** stored in the MBR (Master Boot Record) or EFI partition.

b. Boot Loader (GRUB2 in RHEL)

- RHEL uses **GRUB2 (GRand Unified Bootloader)**.
- GRUB's job:
    1. Present a **boot menu** (choose kernel or recovery mode).
    2. Load the **Linux kernel** (vmlinuz) into memory.
    3. Load the **initramfs** (initial RAM filesystem) which contains drivers/modules needed to mount the real root filesystem.
- After loading, GRUB hands control to the kernel.

---

c. Kernel Initialization

- The kernel (vmlinuz) decompresses and starts executing.
- Loads essential drivers (from **initramfs**) → storage, filesystem, network.
- Mounts the **root filesystem** (/) as defined in /etc/fstab.
- Starts the **init process** (PID 1). In modern RHEL, init = **systemd**.

---

d. systemd Initialization (Replaces init)

- systemd is the **first userspace process** (PID 1).
- Reads configuration from:
    - /etc/systemd/system/
    - /usr/lib/systemd/system/
- Determines the **default target** (like runlevel):
    - multi-user.target (text mode, servers)
    - graphical.target (GUI mode, desktops)
- Brings up required services and daemons in parallel (faster than old SysV init).

---

e. System Services Start

- systemd starts **units** (services, sockets, timers).
- Examples:
    - Networking (NetworkManager.service)
    - Logging (rsyslog.service, systemd-journald)
    - SSH, firewalld, database services, etc.
- Dependencies are resolved automatically.

---

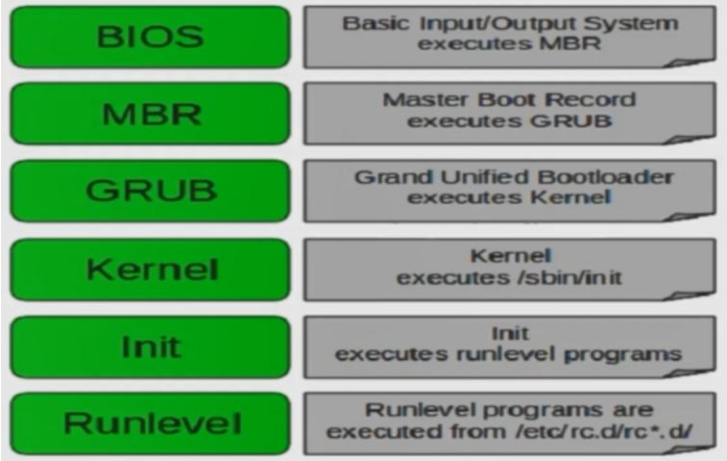
f. Login Prompt

- Finally, you get either:
    - A **text login** (getty) on tty consoles (runlevel 3 / multi-user.target).

- A **graphical login manager** (like GDM) for GUI (runlevel 5 / graphical.target).

At this stage → system is fully booted, and users can log in. ✅

🔹 **Summary Flow**



2.3. Some basic Linux commands

ls – List Directory Contents

Description:

- Displays files and directories in the current or specified directory.

Examples:

ls
ls -l
ls -a
ls /etc
ls -R

Explanation:

- ls → lists files in current directory.
- ls -l → shows detailed info (permissions, owner, size, date).
- ls -a → shows hidden files (those starting with .).
- ls /etc → lists contents of /etc directory.
- ls -R → lists contents **recursively**, including subdirectories.

---

cat – Concatenate and Display Files

Description:

- Shows the content of a file on the terminal.
- Can also join multiple files together.

Examples:

cat file.txt
cat file1.txt file2.txt
cat > newfile.txt

Explanation:

- cat file.txt → displays file content.
- cat file1.txt file2.txt → shows both files one after the other.
- cat > newfile.txt → creates a new file (press Ctrl+D to save).

---

mkdir – Make Directories

Description:

- Creates new directories (folders).

Examples:

mkdir myfolder
mkdir -p /home/student/projects/java

Explanation:

- mkdir myfolder → creates a folder named "myfolder".
- mkdir -p /home/student/projects/java → creates nested directories in one step.

---

tail – Show Last Part of a File

Description:

- Displays the last 10 lines of a file by default. Useful for logs.

Examples:

tail /var/log/messages
tail -n 20 /etc/passwd
tail -f /var/log/secure

Explanation:

- tail file → last 10 lines.
- tail -n 20 file → last 20 lines.

- tail -f file → live view, continuously shows new lines as file updates (useful for monitoring logs).

---

head – Show First Part of a File

Description:

- Displays the first 10 lines of a file by default.

Examples:
head /etc/passwd
head -n 15 /var/log/messages
Explanation:

- head file → first 10 lines.
- head -n 15 file → first 15 lines.

---

6. wc – Word, Line, Character Count

Description:

- Counts words, lines, characters, and bytes in a file.

Examples:

wc file.txt
wc -l file.txt
wc -w file.txt
wc -c file.txt
Explanation:

- wc file.txt → shows lines, words, and bytes.
- wc -l file.txt → counts lines only.
- wc -w file.txt → counts words.
- wc -c file.txt → counts bytes/characters.

---

7. /etc/motd – Message of the Day

Description:

- File that contains a **welcome message** shown to users after login.
- Often used by system admins to show system info, warnings, or announcements.

Examples:

cat /etc/motd
echo "Welcome to RHEL7 Training Server" | sudo tee /etc/motd

Explanation:

- cat /etc/motd → shows the current login message.
- Editing /etc/motd → changes what users see after login.

---

8. history – Command History

Description:

- Shows a list of commands previously executed in the current shell.

Examples:
history
history 20
history | more
!25
!!
Explanation:

- history → lists all commands used so far.
- history 20 → shows the last 20 commands.
- !25 → re-runs the command with history number 25.
- !! → re-runs the **last command**.

✅ **Quick Recap**

- **ls** → list files (-R for recursive)
- **cat** → view file contents
- **mkdir** → make folders
- **tail** → last lines of file
- **head** → first lines of file
- **wc** → count lines/words/chars
- **/etc/motd** → login message
- **history** → recall previous commands

3. Chapter III — File System Hierarchy & Navigation

3.1. File System Hierarchy

Linux follows a **tree-like directory structure** that starts from the **root directory /** (the top of the hierarchy).
Everything in Linux is stored under /.

🔑 **Key Directories in the Root (/) File System:**

Directory	Purpose / What it Stores
/bin	Essential user commands (e.g., ls, cp, mv, cat).
/sbin	System administration commands (e.g., reboot, fdisk).
/etc	Configuration files for the system (e.g., /etc/passwd, /etc/ssh/sshd_config).
/home	Home directories for users (e.g., /home/student).
/root	Home directory for the root (superuser).
/boot	Files needed for booting Linux (kernel, boot loader).
/var	Variable data: logs, emails, spool files (/var/log/messages).
/tmp	Temporary files (cleared after reboot).
/usr	User applications and binaries (like program files).
/lib	Shared libraries (similar to DLLs in Windows).
/media	Mount point for removable devices (USB, CD-ROM).
/mnt	Temporary mount point for filesystems.
/dev	Device files (disks, USB, printers).
/proc	Virtual directory providing process and kernel information.
/srv	Service-related data (e.g., web or FTP data).
/opt	Optional software (third-party apps).

👉 In short: **/etc = configs, /var = logs, /home = users, /bin = commands, /boot = startup files**.

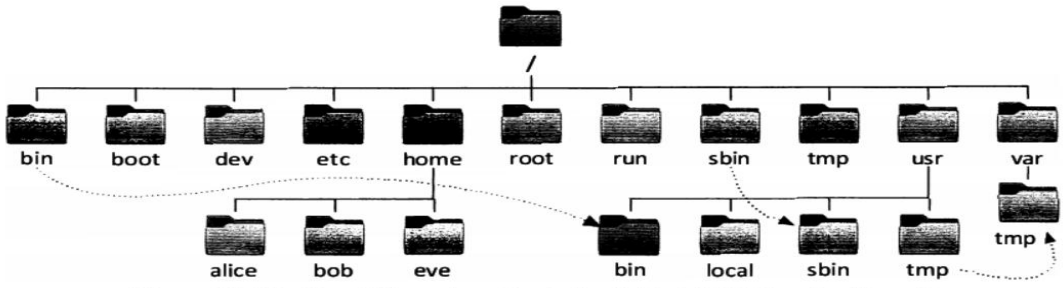


Figure 2.1: Significant file system directories in Red Hat Enterprise Linux 7

```
cd /etc        # go to /etc directory
cd /home/student # go to user's home
cd /           # go to root directory
cd ..          # move one level up
cd -           # switch to previous directory
cd ~           # go to current user's home
```

3.3. Absolute Path

- **Definition:** The complete path from the root / to the file/directory.
- Always starts with /.
- Useful when you want to refer to a file unambiguously, no matter where you are.

Examples:

```
cd /etc/ssh
cd /var/log
```

- /etc/ssh → means go from root → etc → ssh.
- /var/log → root → var → log.

3.4. Relative Path

- **Definition:** Path relative to your **current working directory**.
- Does not start with /.
- Uses . for current directory and .. for parent directory.

Examples:

```
cd documents       # if you are in /home/student, this goes to /home/student/documents
cd ../             # moves one directory up
cd ./myfolder      # access "myfolder" in the current location
```

3.5. ☑ Quick Recap

- **Important directories:**
  - /etc → configuration files
  - /var/log → logs
  - /home → user files
  - /boot → startup files
- **Navigation:**
  - cd → change directory
  - Absolute path → full path from /
  - Relative path → based on current location

4. Chapter IV — Managing Files & Directories

In Linux (RHEL7), file and directory management is mainly done with cp, mv, and rm commands.

4.1. Copying the contents of one file into another

Command:
cp source_file target_file
Example:
cp notes.txt backup_notes.txt
👉 Copies contents of notes.txt into backup_notes.txt.

4.2. Copying one or more files from one location into another

Command:
cp file1 file2 ... destination_directory
Example:
cp file1.txt file2.txt /home/student/Documents/
👉 Copies file1.txt and file2.txt into Documents/.

4.3. Copying one or more directories from one location into another

Use -R (recursive) with cp.
Command:
cp -R dir1 dir2 destination_directory
Example:
cp -R project backup/
👉 Copies entire project directory into backup/.

4.4. Moving the text of one file into another (rename file)

Command:
mv source_file target_file
Example:
mv old.txt new.txt
👉 Renames old.txt to new.txt.

4.5. Moving one or more files from one location to another

Command:
mv file1 file2 ... destination_directory
Example:
mv file1.txt file2.txt /tmp/
👉 Moves file1.txt and file2.txt into /tmp/.

4.6. Moving one or more directories from one location into another

Command:
mv dir1 dir2 destination_directory
Example:
mv project1 project2 /backup/
👉 Moves both directories into /backup/.

4.7. Deleting one or more files

Use rm (remove).
Command:
rm file1 file2 ...
Example:
rm file1.txt file2.txt
👉 Deletes file1.txt and file2.txt.

4.8. Deleting one or more files with -f (force option)

The -f option ignores warnings and forces deletion.
Command:
rm -f file1 file2 ...
Example:
rm -f data.log
👉 Deletes data.log even if write-protected, without asking.

4.9. 9. Deleting one or more directories along with the contents

Use -r (recursive) with rm.
Command:
rm -r dir1 dir2 ...
Example:
rm -r project/
👉 Deletes project/ directory and everything inside.

4.10. Deleting empty directories

Use rmdir (remove directory). Works **only if directory is empty**.

Command:

rmdir dir1 dir2 ...

Example:

rmdir testdir/

👉 Deletes testdir/ only if it has no files inside.

4.11. 📌 Quick Summary Table

Task	Command Example	Notes
Copy file	cp file1 file2	Copies contents
Copy files	cp f1 f2 dir/	Into directory
Copy directory	cp -R dir1 dir2 /path/	Recursive copy
Move file (rename)	mv old new	Renames file
Move files	mv f1 f2 /path/	Moves to dir
Move directory	mv dir1 dir2 /path/	Moves directories
Delete files	rm f1 f2	Removes files
Force delete	rm -f file	No warnings
Delete directory with contents	rm -r dir/	Deletes all inside
Delete empty directory	rmdir dir/	Must be empty

5. Chapter V — Getting Help in Linux

Linux provides several ways to get help, from **keyboard shortcuts** to **manual pages** and **support tools**. Knowing these methods is essential for efficient system management.

5.1. Tab Completion

Description:

- Press **Tab** while typing a command, filename, or directory to **auto-complete** it.

Example:

cd Doc<Tab>

- If there's only one match, it will auto complete to /home/student/Documents/.

5.2. Double Tab (Tab Tab)

Description:

- Press **Tab twice** to see **all possible commands or filenames** that match what you typed.

Example:

ls <Tab><Tab>

- Lists all files/folders in the current directory.

5.3. man – Manual Pages

Description:

- Displays detailed documentation of a command, including **options, syntax, and examples**.

Example:

man ls

- Scroll with arrow keys, q to quit.

Navigating man pages

Command	Result
Spacebar	Scroll forward (down) one screen
PageDown	Scroll forward (down) one screen
PageUp	Scroll backward (up) one screen
DownArrow	Scroll forward (down) one line
UpArrow	Scroll back (up) one line
d	Scroll forward (down) one half-screen
u	Scroll backward (up) one half-screen
/string	Search forward (down) for *string* in the man page
n	Repeat previous search forward (down) in the man page
N	Repeat previous search backward (up) in the man page
g	Go to start of the man page.
G	Go to end of the man page.
q	Exit **man** and return to the command shell prompt

5.4. info

Description:

- Provides **structured documentation**, often more detailed than man.

Example:

info ls

- Navigate with arrow keys, q to quit.

5.5. pinfo

Description:

- Alternative to info with **easier navigation** (like Vim-style interface).

Example:

pinfo ls

- Shows info in sections with easy scrolling and searching.

5.6. whatis

Description:

- Shows **one-line summary** of a command.

Example:

whatis ls

Output:

ls (1) - list directory contents

5.7. Linux Package Documentation Path

Description:

- Documentation for installed packages is usually under:

/usr/share/doc/

Example:

ls /usr/share/doc/vim/

- Contains README, changelogs, and examples.

5.8. Getting Help from Red Hat Support

Description:

- RHEL users can access official support via:
    - Red Hat Customer Portal
    - Documentation, knowledgebase, and technical tickets.

5.9. find

Description:

- Search files or directories **recursively** with conditions.

Example:

find /home/student -name "*.txt"

- Finds all .txt files under /home/student.

5.10. locate

Description:

- Quickly searches files using a **database index** (faster than find).

Example:

locate notes.txt

- Lists all paths containing notes.txt.

5.11. Generating SoS Report

Description:

- sosreport collects system configuration and diagnostic info for **support troubleshooting**.

Example:

sudo sosreport

- Generates a compressed report in /var/tmp/ which can be sent to support teams.

5.12. --help Option

Description:

- Almost all Linux commands support --help for a **quick summary** of options.

Example:

ls --help
cp --help

5.13. whoami

Description:

- Displays the **current logged-in user**.

Example:

whoami
Output:
student

5.14. whereis

Description:

- Shows **binary, source, and manual page locations** for a command.

Example:

whereis ls

Output:

ls: /bin/ls /usr/share/man/man1/ls.1.gz

5.15. pwd – Print Working Directory

Description:

- Displays the **absolute path** of the current directory.

Example:

pwd

Output:

/home/student

5.16. 📌 Quick Reference Table

Command / Tool	Description	Example
Tab	Auto-complete command/file	cd Doc<Tab>
Tab Tab	Show all options	ls <Tab><Tab>
man	Detailed manual	man ls
info	Structured documentation	info ls
pinfo	Alternative info viewer	pinfo ls
whatis	One-line command description	whatis ls
/usr/share/doc/	Package documentation path	ls /usr/share/doc/vim/
Red Hat support	Official technical help	https://access.redhat.com/support
find	Search files recursively	find /home/student -name "*.txt"
locate	Fast search using index	locate notes.txt
sosreport	Generate system report	sudo sosreport
--help	Quick command options	ls --help
whoami	Current user	whoami
whereis	Locate binaries & manuals	whereis ls
pwd	Show current directory	pwd

5.17. Tip for Students:

- Use **man** or **info** for detailed learning.
- Use **--help**, **whatis**, **Tab**, and **Tab Tab** for quick guidance.
- Use **find** and **locate** to efficiently search files.
- sosreport is critical for troubleshooting and sending info to Red Hat support.

6. Chapter VI — vim Editor

6.1. What is an Editor?

Definition:

- An editor is a **software program used to create and modify text files**.
- Editors are essential for writing scripts, configuration files, and documents in Linux.

6.2. Some Common Editors in Linux

Editor	Type
vim	Terminal-based, powerful text editor
nano	Simple terminal editor, beginner-friendly
gedit	GUI-based text editor for desktops
emacs	Advanced text editor with scripting capabilities

6.3. Vim Editor

Description:

- Vim (Vi IMproved) is a **terminal-based editor** in Linux.
- Powerful, with modes and keyboard shortcuts for editing efficiently.

Opening Vim

vim filename.txt

- If the file exists → opens it.
- If not → creates a new file.

6.4. Vim Modes

Mode	Purpose
Normal Mode	Default mode, navigate and issue commands
Insert Mode	Type text into the file (i, a, o)
Visual Mode	Select text for copying, deleting, or formatting (v)
Command Mode	Enter commands like save, quit (:)

6.5. Switching Modes

- i → enter insert mode before cursor
- a → insert after cursor
- o → open new line below and insert
- Esc → return to normal mode
- v → enter visual mode (for selecting text)

6.6. Copy, Cut, Paste

Action	Command
Copy (yank)	yy → copy current line, y in visual mode → copy selection
Cut (delete)	dd → delete current line, d in visual mode → delete selection
Paste	p → paste after cursor, P → paste before cursor

Delete

- x → delete character under cursor
- dw → delete word
- dd → delete entire line

Save and Exit

Command	Description
:w	Save changes
:q	Quit Vim
:wq or ZZ	Save and quit
:q!	Quit without saving

6.7. Gedit Editor

Description:

- gedit is a **GUI-based text editor** for Linux desktops.
- Easier for beginners than Vim.
- Supports syntax highlighting, undo/redo, and multiple tabs.

6.8. Using Gedit

Open Gedit from terminal:

gedit filename.txt &

- & runs it in the background so terminal is free.

Basic Operations:

- **Type text** directly.
- **Copy/Paste:** Ctrl+C / Ctrl+V
- **Cut:** Ctrl+X
- **Save:** Ctrl+S
- **Open new file:** Ctrl+N
- **Exit:** Ctrl+Q

Use Case:

- Editing configuration files
- Writing scripts
- Creating simple text documents

6.9. 📌 Quick Recap

- **Editors** → software to write/edit text.
- **Vim** → terminal-based, powerful, requires modes.
- **Vim Shortcuts** → i, v, yy, dd, p, :wq
- **Gedit** → GUI-based, user-friendly, similar to Notepad.

6.10. ⚡ Tip for Students:

- Use **Vim** for system scripts, config files, and SSH servers.
- Use **Gedit** for GUI editing on your desktop.
- Practice the **Vim shortcuts** in small files to memorize.

7. Chapter VII — Local User & Group Account Management

Managing users and groups is essential for **system security, permissions, and access control** in Linux.

7.1. UID and GID

- **UID (User ID):** Unique number assigned to each user.
- **GID (Group ID):** Unique number assigned to each group.

UID/GID Ranges

Range	Type
0	Root user/group
1–999	System users/groups
1000+	Regular users/groups

Command to show UID/GID
id username
Example:
id student
Output:
uid=1001(student) gid=1001(student) groups=1001(student),10(wheel)

7.2. Important System Files

File	Purpose
/etc/passwd	Stores **user account info** (username, UID, GID, home, shell).
/etc/group	Stores **group info** (group name, GID, members).
/etc/shadow	Stores **encrypted passwords** and password policies.
/home	Home directories of users.

Example of /etc/passwd entry:
student:x:1001:1001:Student User:/home/student:/bin/bash

- Fields: username:x:UID:GID:Full Name:home dir:shell

Default Group Creation

- When a new user is created, a **group with the same name** is automatically created.

7.3. Creating Users

Basic User Creation
sudo useradd username
sudo passwd username

- useradd → creates user
- passwd → sets password

Creating User with Specific UID
sudo useradd -u 1050 student2
sudo passwd student2

7.4. 5. Deleting Users

Delete User Only
sudo userdel username
Delete User with Home Directory
sudo userdel -r username

7.5. 6. Creating Groups

Basic Group Creation
sudo groupadd groupname
Create Group with Specific GID
sudo groupadd -g 1100 developers
Deleting Group
sudo groupdel groupname

7.6. 7. Adding Users to Groups

Add Single User to a Group
sudo usermod -aG groupname username
Add Multiple Users to a Group
sudo usermod -aG groupname user1
sudo usermod -aG groupname user2
Add User to Multiple Groups
sudo usermod -aG group1,group2 username
Create User and Add to Group in One Command
sudo useradd -G groupname username
sudo passwd username

7.7. 8. Giving User Privilege to Perform Root Tasks

useradd user1
passwd user1
Alt+Ctrl+F2 [to open terminal 2] log in with user1
useradd user10 [access denied]
vim /etc/sudoers [gives privileges to user1]
Press I and isert the following line of code
User1 ALL=(ALL) ALL
Press Esc then :wq!
sudo useradd user10 [success]

7.8. 9. Password Policies using chage

- **chage** sets **password expiry, warning, and aging policies**.

Example:
sudo chage -l username # show password info
sudo chage -M 60 username # max days before password expires
sudo chage -m 7 username # min days between password change
sudo chage -W 7 username # warning days before expiration

7.9. 10. Messaging Commands: wall and write

wall – Broadcast Message

- Sends a message to **all logged-in users**.

sudo wall "System maintenance at 10 PM"
write – Send Message to Specific User
write username

- Type your message and press **Ctrl+D** to end.

7.10. 📌 Quick Reference Table

Task	Command Example	Notes
Show UID/GID	id username	Displays user and group IDs
Create user	useradd username	Creates user
Create user with UID	useradd -u 1050 username	Specific UID
Delete user	userdel username	Removes user
Delete user with home	userdel -r username	Removes user + home
Create group	groupadd groupname	Creates group
Create group with GID	groupadd -g 1100 groupname	Specific GID
Delete group	groupdel groupname	Deletes group

Task	Command Example	Notes
Add user to group	usermod -aG groupname username	Append user to group
Add user to multiple groups	usermod -aG group1,group2 username	Multiple groups
Give sudo privileges	Edit /etc/sudoers	Using visudo
Password policies	chage -M 60 username	Set max days etc.
Broadcast message	wall "Message"	All users
Message specific user	write username	Direct message

7.11. Tips for Students:

- Always check /etc/passwd, /etc/group, /etc/shadow for learning purposes.
- Use **id username** to verify UID/GID.
- Test wall and write on practice accounts.

8. Chapter VIII — File and Directory Permissions

In Linux, **every file and directory have permissions** that control who can **read, write, or execute** them.
Permissions are critical for **security and proper access control**. Understanding them allows users and administrators to manage **who can access files and directories** and **what actions they can perform**.
Linux divides permissions into **three categories**:

1. **Owner (user)** – the user who owns the file
2. **Group** – a set of users assigned to a group
3. **Others** – all other users

Effects of permissions on files and directories

Permission	Effect on files	Effect on directories
r (read)	Contents of the file can be read.	Contents of the directory (file names) can be listed.
w (write)	Contents of the file can be changed.	Any file in the directory may be created or deleted.
x (exec)	Files can be executed as commands.	Contents of the directory can be accessed (dependent on the permissions of the files in the directory).

Default Permissions for Files and Directories									
Per	-	-	-	-	-	-	-	-	-
Descrip	Owner			Group Owner			Others		
Example-I									
Per	R	W	-	R	-	-	R	-	-
Binary	1	1	0	1	0	0	1	0	0
Deci	6			4			4		
Final	644 (Default Permission for Files)								
Example-II									
Per	R	W	X	R	-	X	R	-	X
Binary	1	1	1	1	0	1	1	0	1
Deci	7			5			5		
Fina	755 (Default Permissions for Directory)								

8.2. Permission Representation and Calculation

Permissions are shown as **rwx**:

Symbol	Value	Description
r	4	Read
w	2	Write
x	1	Execute

Symbol	Value	Description
-	0	No permission

Example:

-rw-r--r--

- Owner: rw- → 4+2+0 = **6**
- Group: r-- → 4+0+0 = **4**
- Others: r-- → 4+0+0 = **4**
- Numeric representation: **644**

Directories:

drwxr-xr-x

- Owner: rwx → 4+2+1 = 7
- Group: r-x → 4+0+1 = 5
- Others: r-x → 4+0+1 = 5
- Numeric representation: **755**

8.3. Umask Value

- **Umask** determines the **default permission** for new files and directories.
- It **subtracts permissions** from the maximum possible (666 for files, 777 for directories).

Example:

- Umask: 022
- File default: 666-022 = **644**
- Directory default: 777-022 = **755**

View current umask:

umask

Set a new umask:

umask 027

8.4. Changing Permissions with chmod

- **chmod** changes permissions **numerically** or **symbolically**.

Numeric Method

chmod 755 filename

chmod 644 file.txt

Symbolic Method

- + → add permission
- - → remove permission
- = → set exact permission
- **Target**:
  - u → user/owner
  - g → group
  - o → others

- a → all

Examples:

```
chmod u+x script.sh     # Add execute permission for owner
chmod g-w file.txt      # Remove write permission from group
chmod o=r file.txt      # Others can only read
chmod a+r file.txt      # Give read permission to everyone
```

8.5. Changing Ownership

- **chown** changes file **owner** and optionally **group**.
- **chgrp** changes only the group.

Examples:

```
sudo chown student file.txt      # Change owner to student
sudo chown student:developers file.txt   # Change owner and group
sudo chgrp developers file.txt    # Change group only
```

8.6. Quick Reference Table

Task	Command Example	Notes
View permissions	ls -l file.txt	Shows rwx
Numeric chmod	chmod 644 file.txt	Owner rw, group r, others r
Symbolic chmod	chmod u+x file.txt	Add execute to owner
Add permission	chmod g+w file.txt	Add write for group
Remove permission	chmod o-r file.txt	Remove read from others
Set exact permission	chmod a=r file.txt	All can read only
Change owner	chown student file.txt	New owner
Change owner & group	chown student:dev file.txt	Owner and group
Change group	chgrp dev file.txt	Only group

8.7. Tips for Students:

- Files normally **cannot execute by default**; directories can.
- Use **numeric method** for quick changes, **symbolic** for precise adjustments.
- Always check permissions with ls -l.
- Use **umask** to control default permissions for new files/directories.

9. Chapter IX — Linux Processes

In Linux, a **process** is a running instance of a program. Every command you execute, every script running, and system services are all processes.
Understanding **process management** is essential for:

- Monitoring system performance
- Managing CPU and memory usage
- Controlling processes started by users
- Terminating or prioritizing processes

Every process has:

- **PID (Process ID)** – unique identifier
- **PPID (Parent Process ID)** – process that started it
- **Owner/User** – who started the process
- **CPU & Memory usage**
- **State** – running, sleeping, stopped

9.1. Viewing Processes

ps – Process Status

- Displays the processes running on the system.

Command	Description
ps aux	Shows all processes with detailed info (user, PID, CPU, memory, start time, command)
ps -f	Full-format listing of processes
ps -ef	Displays all processes with full details, including PPID
ps -aux	Same as ps aux, BSD-style

Example:
ps aux | grep httpd

- Lists all httpd processes.

9.2. top – Real-time Process Monitor

- Displays **live processes** with CPU, memory usage, PID, and priority.

Example:
top

- Press q to quit.

w -f – Active Users and Terminals

- Shows **who is logged in**, active terminals, and their activities.

Example:

w -f
Output shows: username, terminal, login time, idle time, and command.

9.3. tty – Current Terminal

- Shows the **terminal name** you are currently using.

Example:
tty
Output:
/dev/pts/0

9.4. Killing Processes

kill <PID>

- Terminates a specific process by its PID.

Example:
kill 1234

- Kills process with PID 1234.

---

killall <command-pattern>

- Kills **all processes matching a command name**.

Example:
killall firefox

- Terminates all firefox processes.
- Kill all processes of a specific user:

killall -u user1

---

pkill <pattern>

- Kills processes using a **pattern** in the command name.
- More flexible than killall.

Examples:
pkill httpd # kills all httpd processes
pkill -SIGKILL -u user1 # kills all processes of user1 forcefully
pkill -SIGKILL -t tty3 # kills all processes on terminal 3
pkill -u user1 # kills all processes of user1 except bash shell
pkill -t tty3 # kills all processes on tty3 except bash shell

---

pgrep – List Processes by Pattern

- Finds **processes based on name, user, or terminal**.

Examples:
pgrep -l -u user1 # shows all processes of user1
pgrep -l -t tty3 # shows all processes on terminal 3

9.5. Summary Table of Commands

Command	Purpose	Example
ps aux	List all processes	ps aux
ps -f	Full process info	ps -f
ps -ef	All processes with parent	ps -ef
top	Real-time monitoring	top
w -f	Active users & terminals	w -f
tty	Show current terminal	tty
kill <PID>	Kill a specific process	kill 1234
killall <name>	Kill processes by name	killall firefox
killall -u user	Kill all processes of a user	killall -u user1
pkill <pattern>	Kill processes by pattern	pkill httpd
pkill -SIGKILL -u user	Force kill all processes of user	pkill -SIGKILL -u user1
pkill -SIGKILL -t tty3	Kill all processes on a terminal	pkill -SIGKILL -t tty3
pgrep -l -u user	List all processes of user	pgrep -l -u user1
pgrep -l -t tty3	List all processes of terminal	pgrep -l -t tty3

9.6. Tips for Students:

- Use **ps** for snapshots, **top** for real-time monitoring.
- Use **kill** for individual processes, **killall/pkill** for multiple.
- Always verify processes with **pgrep** before killing.
- Terminal-specific kills (-t tty) avoid accidentally killing system processes.

10. Chapter X — Linux Services and Daemons

Linux systems run **background programs** to provide functionalities like web servers, printing, networking, and scheduled tasks.

- **Service:**
  - A program or process that **provides specific functionality** to users or other programs.
  - Can be started, stopped, or managed manually or automatically at boot.
- **Daemon:**
  - A **background process** that runs continuously, usually without user interaction.
  - Typically ends with **d**, e.g., sshd (SSH daemon).

Key Difference:

Feature	Service	Daemon
Runs in background	Yes	Yes
Provides functionality	Yes	Yes
Controlled by user	Can be	Usually automatic
Interaction	May provide CLI/API	No direct interaction

10.1. Location of Services

- System service unit files are in:

/etc/systemd/system/

- System-wide services:

/usr/lib/systemd/system/

- Legacy init scripts (RHEL6):

/etc/init.d/

10.2. Listing Services and Daemons

RHEL7 (systemd)

```
systemctl list-unit-files --type=service      # Lists all services
systemctl list-units --type=service          # Lists active services
systemctl --failed                           # Lists failed services
```

RHEL6 (init)

```
service --status-all      # List status of all services
chkconfig --list          # Lists enabled/disabled services at runlevels
```

10.3. Managing Services in RHEL7

Action	Command Example	Notes
Start service	sudo systemctl start httpd	Start immediately
Stop service	sudo systemctl stop httpd	Stop immediately
Restart service	sudo systemctl restart httpd	Stop and start
Reload service	sudo systemctl reload httpd	Reload config without stop
Enable at boot	sudo systemctl enable httpd	Auto-start at boot
Disable at boot	sudo systemctl disable httpd	Prevent start at boot
Check status	systemctl status httpd	Active/inactive info
Is-enabled	systemctl is-enabled httpd	Yes/No
Is-active	systemctl is-active httpd	active/inactive
Mask service	sudo systemctl mask httpd	Prevent manual or automatic start
Unmask service	sudo systemctl unmask httpd	Revert mask

10.4. RHEL6 (Legacy commands)

Action	Command
Start service	sudo service httpd start
Stop service	sudo service httpd stop
Restart service	sudo service httpd restart
Reload service	sudo service httpd reload
Enable at boot	sudo chkconfig httpd on
Disable at boot	sudo chkconfig httpd off
Check status	sudo service httpd status

10.5. Disabled vs Stopped Services

Feature	Disabled	Stopped
Startup at boot	No	Can be Yes or No
Current state	May be running or not	Currently not running
Command	systemctl disable service	systemctl stop service
Notes	Prevents automatic start	Stops the process temporarily

10.6. Using Mask Command

- **mask** prevents a service from being started manually or automatically.

sudo systemctl mask httpd
sudo systemctl unmask httpd

- Useful for **disabling critical services** or **prevent accidental start**.

10.7. Quick Recap Table (RHEL7)

Command	Purpose
systemctl start service	Start service immediately
systemctl stop service	Stop service immediately
systemctl restart service	Restart service
systemctl reload service	Reload configuration
systemctl enable service	Enable at boot
systemctl disable service	Disable at boot
systemctl status service	Check status
systemctl is-enabled service	Check if enabled
systemctl is-active service	Check if active
systemctl mask service	Prevent start
systemctl unmask service	Revert mask
systemctl --failed	Show failed services
systemctl list-unit-files --type=service	List all services

10.8. Tips for Students:

- Always use **systemctl status** before starting/stopping to check current state.
- **Masking** is stronger than disabling; even root cannot start it accidentally.
- RHEL6 uses **service** and **chkconfig**, while RHEL7 uses **systemctl**.

11. Chapter XI — Linux Logs

Linux keeps a variety of **log files** to record system activities, errors, security events, boot information, mail server activity, and more.

11.1. Why logs are important:

- Troubleshooting system and application issues
- Monitoring user activity and security
- Auditing system performance
- Tracking errors and warnings

Logs help administrators maintain system health and prevent failures.

11.2. Log File Locations

- Most logs are stored in the directory:

/var/log

- Examples of key logs:

Log File	Purpose
/var/log/messages	General system messages and most syslog errors
/var/log/secure	Security and authentication logs (login attempts, sudo usage)
/var/log/maillog	Mail server logs
/var/log/boot.log	Boot process messages

11.3. Viewing Logs Using cat

- Display entire log files:

```
cat /var/log/messages
cat /var/log/secure
cat /var/log/maillog
cat /var/log/boot.log
```

- View only **head or tail** of logs:

```
tail -n 10 /var/log/secure   # Last 10 lines
head -n 5 /var/log/boot.log  # First 5 lines
```

11.4. Viewing Logs Using journalctl

journalctl displays logs collected by **systemd journal**.

Command	Description
journalctl	Show all logs
journalctl -n 5	Show last 5 log entries

Command	Description
journalctl -f	Follow logs in real-time (like tail -f)
journalctl -p err	Show only **error** logs
journalctl -p warning	Show **warning** logs
journalctl -p crit	Show **critical** logs
journalctl -p emerg	Show **emergency** logs
journalctl --since today	Show logs from today
journalctl --since "2018-01-01"	Show logs since specific date
journalctl -b	Show logs since last boot

11.5. Date and Time Related Commands

- Logs often rely on accurate system date and time. Use timedatectl to manage time:

Command	Purpose
timedatectl	Shows current date, time, and NTP status
timedatectl list-timezones	Shows available timezones
timedatectl set-time "YYYY-MM-DD HH:MM:SS"	Sets system time
timedatectl set-ntp true	Enables automatic time synchronization via NTP

11.6. Tips for Students

- Use tail -f or journalctl -f to **monitor logs in real-time**.
- Use **priority filtering** (-p) in journalctl to focus on critical logs.
- Combine grep with log files to search for specific events, e.g.:

grep "Failed password" /var/log/secure

- Accurate **time settings** ensure log timestamps are correct for troubleshooting.

11.7. Quick Reference Table

Task	Command
View all system logs	journalctl
Last 5 logs	journalctl -n 5
Error logs	journalctl -p err
Warning logs	journalctl -p warning
Emergency logs	journalctl -p emerg
Logs since last boot	journalctl -b
View boot log	cat /var/log/boot.log

Task	Command
View mail log	cat /var/log/maillog
View secure log	cat /var/log/secure
Monitor logs live	journalctl -f
Set system time	timedatectl set-time "YYYY-MM-DD HH:MM:SS"
Enable NTP	timedatectl set-ntp true

11.8. Tip for Students:

- Logs are your **first tool for troubleshooting**. Learn to filter, search, and follow them efficiently.
- Always check **secure logs** for authentication and sudo activity.
- Combine journalctl with grep and tail for powerful monitoring.

12. Chapter XII — Yum Server

YUM (Yellowdog Updater, Modified) is the **default package manager for RHEL and its derivatives**. It is used to:

- Install new software packages
- Update existing packages
- Remove or uninstall software
- Manage dependencies automatically

Importance of YUM:

- Simplifies package management
- Handles dependencies automatically
- Allows installation from local or remote repositories
- Tracks installed packages and their history

12.1. Step 1: Copy RHEL DVD to Local Directory

1. Create a directory to store ISO:

```
mkdir /iso
```

2. Copy contents from DVD to ISO file:

```
dd if=/dev/cdrom of=/iso/rh7.iso
```

- Copies the RHEL DVD to /iso/rh7.iso.

3. Verify ISO file:

```
cd /iso
ls -l
```

12.2. Step 2: Mount the ISO

1. Create a mount point:

```
mkdir /Packages
```

2. Mount the ISO to /Packages:

```
mount -t iso9660 -o loop /iso/rh7.iso /Packages
```

3. Make the mount persistent:

- Edit /etc/fstab:

```
vim /etc/fstab
```

- Add the line:

```
/iso/rh7.iso   /Packages   iso9660    user,loop   0 0
```

12.3. Step 3: Configure Local YUM Repository

1. Navigate to YUM repos directory:

```
cd /etc/yum.repos.d/
```

2. Create repository file:

vim rh7.repo

- Add the following content:

```
[rh7]
name=rh7
baseurl=file:///Packages
enabled=1
gpgcheck=0
```

- Save and exit (:wq).

12.4. Step 4: Disable RHN Plugin (Optional)

- Navigate to plugin configuration:

```
cd /etc/yum/pluginconf.d
vim rhnplugin.conf
```

- Set:

```
enabled=0
```

12.5. Step 5: Clean and Update YUM

1. Clean YUM cache:

```
yum clean all
```

2. Update YUM server:

```
yum update
```

12.6. Step 6: Installing Packages Using YUM

- Install a package:

```
yum install -y <package_name>
```

Examples:

```
yum install -y bind*      # Install DNS server with all related packages
yum install -y http*      # Install Apache HTTP server
yum install -y samba*     # Install Samba server
```

12.7. Step 7: Managing Installed Packages

Command	Description
yum repolist	List all available packages
yum list yum*	List installed packages
yum groups list	Show package groups/features
yum info httpd	Show detailed info about a package
yum provides /var/www/html	Find package providing a file/path
yum remove -y http*	Uninstall HTTP server and related packages

Command	Description
yum history	Show installation/uninstallation history

12.8. Step 8: Updating Kernel and System

- Check kernel version:

```
uname -r      # Shows kernel version
uname -a      # Shows kernel version with date/time
```

- Update system if registered:

```
yum update
```

12.9. Step 9: Enabling/Disabling Local Repo and RHN Plugin

- Enable RHN plugin:

```
cd /etc/yum/pluginconf.d
vim rhnplugin.conf
enabled=1
```

- Disable local repo if needed:

```
cd /etc/yum.repos.d/
vim rh7.repo
enabled=0
```

- Clean and update YUM again:

```
yum clean all
yum update
```

12.10. Tips for Students

- Use -y with YUM to automatically answer yes to prompts.
- Always run yum clean all after changing repository configurations.
- Use wildcards (*) to install multiple related packages.
- Keep /iso and /Packages accessible if using local YUM repo.
- Check package installation status with yum history.

13. Chapter XIII — Basic Networking

Networking is essential for connecting your RHEL system to other devices and the internet. Linux provides tools for:

- Configuring **IP addresses** (static and dynamic)
- Setting **hostnames**
- Managing **network interfaces (NICs)**
- Managing **routing, DNS, and connectivity**
- Controlling **firewall and SELinux security**

RHEL 7 uses **NetworkManager** (nmcli) for managing network connections, along with traditional tools like ip, ping, and configuration files in /etc/sysconfig/network-scripts.

13.1. Network Configuration Files

- **Location:**

/etc/sysconfig/network-scripts/

- Each interface has a config file (e.g., eno1, eno2)

```
cd /etc/sysconfig/network-scripts
ls
vim eno-16777736
```

- You can modify IP addresses, gateways, DNS, and autoconnect options.

13.2. Hostname Management

- Set hostname:

```
hostnamectl set-hostname Server1
```

- Check hostname:

```
hostnamectl status
hostnamectl
cat /etc/hostname
```

- Changes may require reboot:

```
systemctl reboot
```

13.3. Viewing Network Information

Command	Purpose
ip addr / ip addr show	Show IP addresses and network info
ip link	Show network interfaces and their status
nmcli dev status	Show NIC status
nmcli connection show	Show configured network connections
ip route	Check routing table

Command	Purpose
nmcli device show eno2	Show detailed connection info including DNS and gateway

13.4. Managing Network Connections (nmcli)

- Create a new connection with static IP:

```
nmcli connection add con-name Test ifname eno-2 type Ethernet ip4 192.168.10.1/24 gw4
192.168.10.10
```

- Modify an existing connection:

```
nmcli connection modify Test ipv4.dns 8.8.8.8       # Add DNS server
nmcli connection modify eno2 ipv4.address 10.0.0.1/8 # Change IPv4 address
nmcli connection eno2 ipv4.method auto              # Set dynamic IP
```

- Connect or disconnect a connection:

```
nmcli connection up Test2
nmcli connection down Test2
nmcli device connect eno16777738
```

- Delete a connection:

```
nmcli connection del Test
```

- Enable autoconnect with dynamic IP:

```
nmcli connection add con-name Test ifname eno2 type Ethernet autoconnect yes
```

13.5. Testing Network Connectivity

- Ping a host:

```
ping 192.168.20.1
ping -c 10 192.168.1.1   # Send 10 packets
```

- Trace the path to a remote host:

```
tracepath google.com
```

13.6. Firewall Management

- Check firewall status (SELinux enforcement):

```
getenforce         # Shows current SELinux mode
setenforce 1       # Turn on enforcing
setenforce 0       # Turn off enforcing
```

- Stop, start, disable, enable firewall service (RHEL7):

```
systemctl stop iptables
systemctl start iptables
systemctl disable iptables
systemctl enable iptables
```

- Permanent SELinux setting:

```
vim /etc/SELinux/config
```

SELINUX=enforcing # or permissive

- RHEL6 alternative commands:

service iptables stop
service iptables start
chkconfig iptables off

13.7. Quick Tips for Students

- Always verify IP configuration using ip addr or nmcli device show.
- Use ping and tracepath to troubleshoot connectivity issues.
- Dynamic IP addresses use DHCP (ipv4.method auto), static IPs are manually configured.
- Hostname changes may require reboot for some services.
- SELinux and firewall settings can block connectivity; ensure proper configuration.

13.8. Summary Table of Useful Commands

Task	Command
Show IP address	ip addr
Show NIC status	nmcli dev status
Show routing table	ip route
Create static IP connection	nmcli connection add con-name Test ifname eno2 type Ethernet ip4 192.168.10.1/24 gw4 192.168.10.10
Modify DNS	nmcli connection modify Test ipv4.dns 8.8.8.8
Connect/Disconnect NIC	nmcli connection up Test2 / nmcli connection down Test2
Delete connection	nmcli connection del Test
Check hostname	hostnamectl status
Set hostname	hostnamectl set-hostname Server1
Ping host	ping 192.168.1.1
Trace route	tracepath google.com
Check SELinux	getenforce
Set SELinux enforcing	setenforce 1
Stop firewall	systemctl stop iptables
Disable firewall at boot	systemctl disable iptables

Tip:

Combining nmcli, ip, ping, and tracepath allows students to fully manage and troubleshoot network configurations in RHEL 7.

14.Chapter XIV — Linux Firewall

Linux firewalls are essential for securing systems by **controlling network traffic**. In RHEL 7, the default firewall is **Firewalld**, replacing traditional **iptables** from earlier versions.

Key Points:

Feature	iptables	Firewalld	SELinux
Purpose	Packet filtering and firewall rules	Dynamic firewall management with zones and services	Mandatory Access Control for system security
Operation	Static rules, manual configuration	Zones, services, ports, runtime & permanent changes	Controls access to files, processes, ports, and resources
Management	Command-line (iptables)	CLI (firewall-cmd) and GUI (firewall-config)	Configuration files & SELinux commands
RHEL Version	RHEL 6 and earlier	RHEL 7	RHEL 6/7/8

Focus of this chapter: Firewalld and firewall management on RHEL 7.

14.1. Check Firewall Installation and Status

```
hostname                # Show hostname
ip addr show            # Show network interfaces & IP
rpm -qa | grep firewalld    # Check if firewalld is installed
rpm -qc firewalld       # Show configuration files
vim /etc/firewalld/firewalld.conf  # Edit main config file if required
systemctl status firewalld  # Show firewalld service status
firewall-cmd --state    # Check firewall runtime state
```

14.2. Zones and Services

- Default zone: **public**

```
firewall-cmd --get-default-zone
firewall-cmd --list-all
```

- List available zones:

```
firewall-cmd --get-zones
```

- List services:

```
firewall-cmd --get-services
```

- Services are defined in:

/usr/lib/firewalld/services/
Example:
cd /usr/lib/firewalld/services
ls
cat ssh.xml
cat http.xml
cat mysql.xml

14.3. Adding/Removing Ports and Services

- Add a TCP port (runtime):

firewall-cmd --add-port=3306/tcp

- Remove a TCP port (runtime):

firewall-cmd --remove-port=3306/tcp

- Add/remove a service (runtime):

firewall-cmd --add-service=mysql
firewall-cmd --remove-service=mysql

⚠️ **Important:** Without --permanent, changes are lost after firewall-cmd --reload.

- Make changes permanent:

firewall-cmd --permanent --add-service=mysql
firewall-cmd --reload

- Add/remove multiple ports/services:

firewall-cmd --add-port={3306/tcp,5000/tcp,5500/tcp}
firewall-cmd --add-port=3000-4000/tcp # Range of ports
firewall-cmd --add-service={mysql,http,ftp,dns}

14.4. Forwarding Ports

- Redirect incoming traffic from one port to another:

netstat -nltp # Check which ports/services are used
firewall-cmd --add-forward-port=port=5000:proto=tcp:toport=22
firewall-cmd --list-all

14.5. Rich Rules

- Rich rules allow **advanced and dynamic firewall configuration**:

firewall-cmd --add-rich-rule='rule family="ipv4" source address="10.0.0.2" accept'
firewall-cmd --add-rich-rule='rule family="ipv4" source address="10.0.0.3" drop'
firewall-cmd --list-rich-rules # Show rich rules

14.6. Panic Mode

- Drops **all incoming/outgoing traffic** temporarily:

```
firewall-cmd --panic-on
firewall-cmd --query-panic
firewall-cmd --panic-off
```

14.7. Assigning Interfaces to Zones

- Check which zone an interface is in:

```
firewall-cmd --get-zone-of-interface eno16777736
```

- List interfaces in a zone:

```
firewall-cmd --zone=public --list-interfaces
```

- Move an interface to another zone:

```
firewall-cmd --zone=internal --change-interface=eno16777736
```

14.8. Default Zone Management

- Show default zone:

```
firewall-cmd --get-default-zone
```

- Change default zone:

```
firewall-cmd --set-default-zone=internal
firewall-cmd --set-default-zone=public
```

14.9. Firewalld vs Iptables

- Firewalld uses **zones and services** for dynamic management
- Iptables uses **static rules** and is less flexible
- **Only one** can run at a time:
  - Starting firewalld stops iptables automatically
  - Starting iptables stops firewalld automatically

14.10. GUI Firewall Management

- Access GUI:

Alt + F2 → firewall-config

- Provides visual interface for zones, services, ports, and rules.

14.11. Quick Reference Table

Task	Command
Check hostname	hostname
Show IP	ip addr show

Task	Command
Check firewalld	systemctl status firewalld
Show firewall state	firewall-cmd --state
List default zone	firewall-cmd --get-default-zone
List all firewall rules	firewall-cmd --list-all
Add port	firewall-cmd --add-port=3306/tcp
Remove port	firewall-cmd --remove-port=3306/tcp
Add service	firewall-cmd --add-service=mysql
Remove service	firewall-cmd --remove-service=mysql
Add multiple ports	firewall-cmd --add-port={3306/tcp,5000/tcp}
Add port range	firewall-cmd --add-port=3000-4000/tcp
Rich rule	firewall-cmd --add-rich-rule='rule family="ipv4" source address="10.0.0.2" accept'
Panic mode ON/OFF	firewall-cmd --panic-on / --panic-off
Assign interface to zone	firewall-cmd --zone=internal --change-interface=eno16777736

14.12. Tip for Students:

- Use **permanent option** to save changes.
- Always verify which zone your interface is assigned to before adding rules.
- Rich rules enable granular access control based on IP, protocol, or source.
- Remember: **firewalld and iptables cannot run simultaneously**.

15. Chapter XV — DNS Server

DNS (Domain Name System) is a critical network service that translates **human-readable domain names** (e.g., server1.kabul.com) into **IP addresses** (e.g., 192.168.50.2). Without DNS, users would have to remember numeric IP addresses to access services.

Requirements for a DNS server on RHEL 7:

1. A configured **network interface** with a static IP.
2. A **YUM repository** to install necessary DNS packages (bind).
3. Firewall configuration allowing **DNS traffic (port 53)**.
4. Proper **zone files** for forward and reverse lookups.

15.1. Network Preparation

- Configure a static IP using nmcli:

```
nmcli connection add con-name Test ifname eno-2 type Ethernet ip4 192.168.10.1/24
nmcli device connect eno216777738
nmcli connection up Test
```

- Ensure your **YUM server is configured** for installing packages.

15.2. Install and Verify DNS Packages

- Check if bind (DNS service) is installed:

```
rpm -qa | grep bind
```

- Install DNS packages:

```
yum -y install bind*
```

- Check service status:

```
systemctl status named
```

15.3. Firewall Configuration for DNS

- Stop iptables and start firewalld:

```
systemctl stop iptables
systemctl mask iptables
systemctl start firewalld
systemctl status firewalld
firewall-cmd --state
```

- Allow DNS service through firewall permanently:

```
firewall-cmd --permanent --add-service=dns
firewall-cmd --reload
firewall-cmd --list-all
```

15.4. Configure DNS Service (named)

- Start and enable named service:

systemctl start named
systemctl enable named

- Edit the main configuration file:

vim /etc/named.conf

Example configuration:

```
zone "test.com" IN {
   type master;
   file "test.com.forward";
};
zone "10.168.192.in-addr.arpa" IN {
   type master;
   file "test.com.reverse";
};

listen-on port 53 { any; };
allow-query { any; };
```

Note: Replace 127.0.0.1 with any to allow DNS queries from all clients.

15.5. Forward and Reverse Zone Files

- Go to the zone files directory:

cd /var/named/
ls

- Copy default templates:

cp -rf named.localhost kabul.com.forward
cp -rf named.loopback kabul.com.reverse

- Change ownership:

chown root:named kabul.com.*

Forward Zone File Example (kabul.com.forward)

```
$TTL 1D
@       IN SOA  @ root.learnitguide.net. (
                                        0   ; serial
                                        1D  ; refresh
                                        1H  ; retry
                                        1W  ; expire
                                        3H ) ; minimum
NS      @
A       192.168.10.1
server1 A       192.168.50.2
cl1     A       192.168.50.10
```

Reverse Zone File Example (kabul.com.reverse)

```
$TTL 1D
@   IN SOA @ rname.invalid. (
            0   ; serial
            1D  ; refresh
            1H  ; retry
            1W  ; expire
            3H ) ; minimum
NS  @
A   192.168.50.2
AAAA    ::1
2   PTR server1
10  PTR cl1
```

15.6. Validate and Restart DNS

- Check configuration syntax:

```
named-checkconf
```

- Restart DNS service:

```
systemctl restart named
```

15.7. Test DNS Resolution

- Use nslookup and ping:

```
nslookup server1.kabul.com
ping server1.kabul.com
```

15.8. Configure Resolver on Clients

- Edit /etc/resolv.conf to point to your DNS server:

```
vim /etc/resolv.conf
```

Example content:

```
domain kabul.com
nameserver 192.168.100.1
```

- Restart networking:

```
systemctl restart network
```

- Test DNS resolution again:

```
nslookup server1.kabul.com
ping server1.kabul.com
```

15.9. Summary

- **DNS** maps domain names to IPs, enabling easier network navigation.
- **Bind** (named) is the DNS server package on RHEL.

- **Firewall rules** must allow port 53 (TCP/UDP) for DNS.
- Zone files define **forward** and **reverse** mappings.
- Always check DNS configuration with named-checkconf and test with nslookup or ping.

15.10. Tip for Students:

- Always make **backups of zone files** before editing.
- Use consistent naming conventions for forward/reverse zones.
- Check firewall and SELinux rules to ensure DNS traffic is not blocked.

16. Chapter XVI — DHCP Server

DHCP (Dynamic Host Configuration Protocol) is a network service that **automatically assigns IP addresses, subnet masks, gateways, and DNS information** to clients on a network.

Importance of DHCP:

1. **Automatic IP Assignment** – Clients do not need manual IP configuration.
2. **Reduces Errors** – Eliminates IP conflicts caused by manual assignment.
3. **Centralized Management** – IP addresses are managed from a single server.
4. **Efficient IP Utilization** – Recycles unused IP addresses to other devices.
5. **Supports Large Networks** – Useful in enterprise environments where manual IP management is impractical.

16.1. Network Preparation

- Assign a static IP to the DHCP server:

```
nmcli connection add con-name Test ifname eno-2 type Ethernet ip4 192.168.10.1/24
nmcli device connect eno216777738
nmcli connection up Test
```

- Ensure **YUM repository** is configured for package installation.

16.2. Install DHCP Packages

- Check if DHCP is installed:

```
rpm -qa | grep dhcp
```

- Install DHCP server packages:

```
yum -y install dhcp*
```

- Check the DHCP service status:

```
systemctl status dhcpd
```

16.3. Firewall Configuration

- Stop iptables if running:

```
systemctl stop iptables
systemctl mask iptables
```

- Start firewalld and check its status:

```
systemctl start firewalld
systemctl status firewalld
systemctl status iptables
```

16.4. Configure DHCP Server

Global Network Configuration

Edit /etc/dhcp/dhcpd.conf to define the IP range and network parameters:

vim /etc/dhcp/dhcpd.conf

```
subnet 192.168.100.0 netmask 255.255.255.0 {
 range 192.168.100.100 192.168.100.200;
 option domain-name-servers 8.8.8.8;
 option routers 192.168.100.1;
 option broadcast-address 192.168.100.255;
 default-lease-time 600;
 max-lease-time 7200;
{
```

Fixed IP Assignment for Specific Host

vim /etc/dhcp/dhcpd.conf

```
host test-serverb {
 hardware ethernet 00:0C:29:F9:23:4F;
 fixed-address 192.168.100.150;}
```

Explanation:

- subnet defines the network range for dynamic IP allocation.
- range specifies the pool of IPs available to clients.
- option routers sets the default gateway.
- option domain-name-servers specifies DNS servers.
- host section allows assigning a fixed IP to a device using its MAC address.

16.5. Start and Manage DHCP Service

```
systemctl start dhcpd
systemctl stop dhcpd
systemctl restart dhcpd
chkconfig dhcp on
```

- Verify configuration syntax:

```
dhcpd -t
```

16.6. Configure Clients for DHCP

- Edit the network interface configuration to use DHCP:

```
vim /etc/sysconfig/network-scripts/ifcfg-eno16777736
```

- Set:

```
BOOTPROTO=dhcp
```

- Add connection with autoconnect:

```
nmcli connection add con-name test ifname eno16777736 type Ethernet autoconnect yes
```

- Restart network service:

```
systemctl restart network
```

16.7. Verify DHCP Operation

- Check assigned IP addresses on clients:

ip addr show

- View active DHCP leases on the server:

cat /var/lib/dhcpd/dhcpd.leases

- Test connectivity:

ping 192.168.100.1

16.8. Summary

- DHCP automates IP address management and reduces network errors.
- Server configuration is done in /etc/dhcp/dhcpd.conf.
- Fixed IP addresses can be assigned to specific clients via MAC addresses.
- Firewall must allow DHCP traffic (port 67/68 UDP).
- Verify configuration with dhcpd -t and monitor leases in /var/lib/dhcpd/dhcpd.leases.

17. Chapter XVII — Web Server

A **Web Server** is a software application that serves **web content (HTML, CSS, JavaScript, images, etc.) to clients** (browsers) over **HTTP/HTTPS protocols**.

Importance & Requirements of a Web Server:

1. **Serve Web Pages** – Delivers content to client browsers.
2. **Host Multiple Sites** – Supports virtual hosts to serve multiple domains.
3. **Supports Dynamic Content** – Works with scripting languages like PHP, Python.
4. **Security** – Integrates with firewalls and SSL/TLS certificates.

Requirements:

- RHEL 7 Server with static IP.
- YUM repository configured.
- Firewall properly configured to allow HTTP (80) and HTTPS (443) services.

17.1. Network Preparation

Assign a static IP to the server:

nmcli connection add con-name Test ifname eno-2 type Ethernet ip4 192.168.10.1/24

nmcli device connect eno216777738

nmcli connection up Test

Ensure YUM repository is already configured.

17.2. Install Apache HTTP Server

- Check if HTTP packages are installed:

rpm -qa | grep http

systemctl status httpd

- Install HTTP packages:

yum -y install http*

- Manage firewall to allow HTTP/HTTPS:

systemctl stop iptables

systemctl mask iptables

systemctl start firewalld

systemctl status firewalld

firewall-cmd --state

firewall-cmd --permanent --add-service=http

firewall-cmd --permanent --add-service=https

firewall-cmd --reload

firewall-cmd --list-all

17.3. Start & Enable Web Service

systemctl start httpd

systemctl enable httpd

- Verify HTTPD process and configuration:

apachectl configtest
httpd -T

17.4. Create Web Content

- Create directory for website:

mkdir /var/www/RU
cd /var/www/RU
vim home.html

- Add content to home.html:

Welcome to RANA University

- Restart Apache after changes:

systemctl restart httpd

17.5. Configure Virtual Host

Edit /etc/httpd/conf/httpd.conf and add:

```
<VirtualHost 192.168.10.1>
  DocumentRoot /var/www/RU
  ServerName www.RU.com
  DirectoryIndex home.html
</VirtualHost>
```

- Restart and enable Apache:

systemctl restart httpd
systemctl enable httpd

17.6. Configure /etc/hosts File

- Edit /etc/hosts on server and client:

vim /etc/hosts
Add:
192.168.10.1 www.RU.com

- Save and restart HTTPD:

systemctl restart httpd
systemctl enable httpd

17.7. Testing

- Open a browser and browse:

http://www.RU.com

- The page should display:

Welcome to RANA University

17.8. Summary

- Web servers allow hosting websites on RHEL 7.
- Apache HTTPD is the default web server for RHEL.

- Network, firewall, and YUM configuration are prerequisites.
- Virtual hosts allow multiple domains on a single server.
- /etc/hosts ensure proper hostname resolution for testing.

18. Chapter XVIII — Web Server

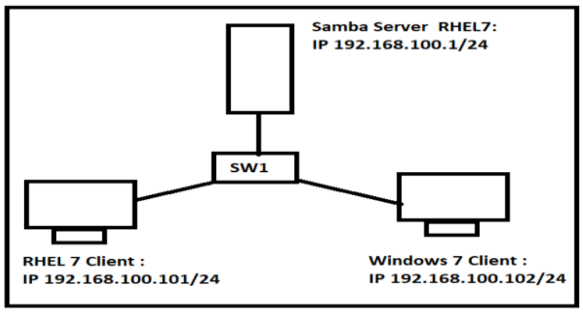
Samba is open-source software that allows **Linux/Unix systems to share files and printers with Windows clients** using the **SMB/CIFS protocol**.

Importance:

1. Enables cross-platform file sharing between Linux and Windows.
2. Allows centralized storage and access for multiple clients.
3. Supports user authentication and access control.
4. Integrates with network services like DNS and DHCP for smoother management.

Requirements:

- RHEL 7 Server with static IP.
- YUM repository configured.
- Firewall configured to allow Samba services.
- SELinux is properly configured for shared directories.



18.1. Network Preparation

Set up static IP for server:
nmcli connection add con-name Test ifname eno-2 type Ethernet ip4 192.168.10.1/24
nmcli device connect eno216777738
nmcli connection up Test
Ensure YUM repository is already configured.

18.2. Install Samba Server

- Check if Samba is installed:

rpm -qa | grep samba
systemctl status smb

- Install Samba packages:

yum -y install samba*

- Manage firewall:

systemctl stop iptables
systemctl mask iptables
systemctl start firewalld
systemctl status firewalld

18.3. Configure Shared Directory

- Create and set permissions:

```
mkdir /SambaSharedData
chmod -R 777 /SambaSharedData
```

- Allow Samba through firewall:

```
firewall-cmd --list-all
firewall-cmd --permanent --add-service=samba
firewall-cmd --reload
firewall-cmd --list-all
```

18.4. Configure Samba

Edit /etc/samba/smb.conf and add:

```
[SambaSharedData]
   comment = Samba Share
   path = /SambaSharedData
   public = yes
   writable = yes
   printable = no
   valid users = root
   browseable = yes
   host allow = 192.168.30.0
```

- Set SELinux context:

```
chcon -t samba_share_t /SambaSharedData
```

- Add root user to Samba:

```
smbpasswd -a root
```

- Start and enable Samba services:

```
systemctl start smb
systemctl enable smb
```

- Verify configuration:

```
testparm
```

18.5. Accessing Samba Share from RHEL Client

- Configure client network:

```
nmcli connection add con-name Test ifname eno16777736 type Ethernet ip4
192.168.100.101/24
nmcli device connect eno16777736
nmcli connection up Test
systemctl restart network
ip addr
ping 192.168.100.1
```

- Install Samba client and access share:

yum -y install samba-client*
smbclient -L 192.168.100.1 -U root
smbclient //192.168.100.1/SambaSharedData -U root
You need the **root username and password** created on the Samba server.

18.6. Accessing Samba Share from Windows Client

1. Configure IP address on Windows client.
2. Check connectivity with server: ping 192.168.100.1.
3. Access share:

Run => \\192.168.100.1

4. Enter **root username and password** created on the Samba server.

18.7. Summary

- Samba enables **cross-platform file sharing** between Linux and Windows.
- Proper **network, firewall, SELinux, and user configuration** are required.
- Accessing Samba shares can be done using both **Linux clients** (via smbclient) and **Windows clients** (via network path).
- Permissions and firewall rules ensure security and controlled access.

19. Chapter XIX — vsFTP Server

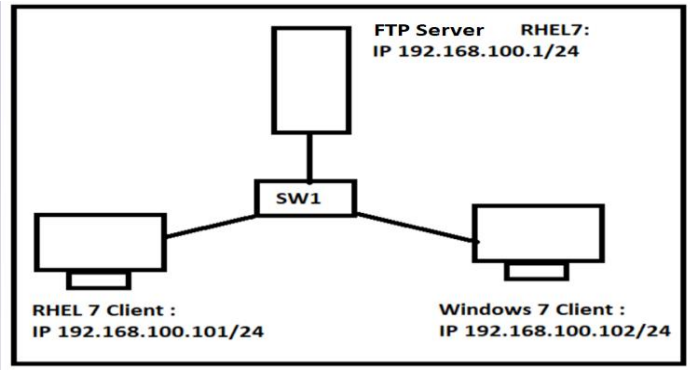
vsFTP (Very Secure FTP) is a secure FTP server implementation for Linux that provides **fast and safe file transfers** over the network.

Importance:

1. Allows uploading and downloading of files between client and server.
2. Supports both **anonymous** and **authenticated** users.
3. Integrates with firewall and SELinux for secure access.
4. Essential for managing file transfers in enterprise networks.

Requirements:

- RHEL 7 server with static IP.
- YUM repository configured.
- Firewall configured to allow FTP services.
- SELinux configured properly if enabled.



19.1. Network Preparation

Set up IP and check connectivity:

nmcli connection add con-name Test ifname eno-2 type Ethernet ip4 192.168.100.1/24

nmcli device connect eno216777738

nmcli connection up Test

Ensure YUM repository is configured.

19.2. Install vsFTP Server

- Check if vsftpd is installed:

rpm -qa | grep vsftpd

yum list vsftpd

systemctl status vsftpd

- Install vsftpd packages:

yum -y install vsftpd*

- Manage firewall:

systemctl stop iptables
systemctl mask iptables
systemctl start firewalld
systemctl status firewalld

19.3. Configure FTP Directory

- Default FTP directory: /var/ftp/pub

chmod -R 777 /var/ftp/pub

- Allow FTP through firewall:

firewall-cmd --list-all
firewall-cmd --permanent --add-service=ftp
firewall-cmd --permanent --add-port=21/tcp
firewall-cmd --reload
firewall-cmd --list-all

19.4. Configure vsFTP Server

Edit /etc/vsftpd/vsftpd.conf:

1. Uncomment and modify:

anon_upload_enable=YES
ascii_upload_enable=YES
ascii_download_enable=YES
anonymous_enable=YES # Or NO for non-anonymous FTP

2. Save and exit: :wq

- Start and enable vsftpd service:

systemctl start vsftpd
systemctl enable vsftpd

19.5. Accessing FTP Server from Linux Client

- Test FTP connection:

ftp 192.168.100.1
ftp localhost
ftp 127.0.0.1

- FTP commands:

ls # list files
cd pub # change directory
get file1 # download file

put file2 # upload file
exit or bye # disconnect

- Install FTP client if missing:

yum -y install ftp

19.6. Accessing FTP Server from Windows Client

- Open cmd or browser:

ftp 192.168.100.1
or
ftp://192.168.100.1

- Enter **username**: ftp or anonymous.
- Enter **password**: (if required).
- FTP commands: cd pub, get file1.txt, put file2.txt, exit.

19.7. Non-Anonymous FTP Access

- Create user accounts on FTP server.
- Use created accounts for authentication while accessing FTP from clients.

19.8. Secure File Copy between Linux Systems

- Using scp command:

```
# Copy local file to remote server
scp /home/user/file1 192.168.10.1:/home

# Copy file from remote server to local system
scp 192.168.10.1:/home/user/file1 /home
```

19.9. Summary

- vsFTP is a **secure, efficient FTP solution** for Linux.
- Supports **anonymous and authenticated access**.
- Integrated with **firewall and SELinux** for secure file transfers.
- ftp and scp commands provide flexible ways to transfer files between Linux and Windows/Linux clients.

20. Chapter XX — NTP Server

NTP (Network Time Protocol) is a protocol used to **synchronize the clocks of computers over a network**.

Importance:

1. Ensures consistent timestamps across servers and clients.
2. Critical for logging, security, and scheduled tasks.
3. Supports hierarchical time sources using **stratum levels**.
4. Helps avoid issues caused by time drift on servers.

Requirements:

- RHEL 7 server with static IP.
- YUM repository configured.
- Firewall configured to allow NTP service.
- Clients (Linux or Windows) connected to the network.

20.1. Network Preparation

Set IP and check connectivity:

nmcli connection add con-name Test ifname eno-2 type Ethernet ip4 192.168.100.1/24

nmcli device connect eno216777738

nmcli connection up Test

Ensure YUM repository is configured.

20.2. Install NTP Packages

- Check if NTP is installed:

rpm -qa | grep ntp

yum list ntp

systemctl status ntpd

- Install NTP:

yum -y install ntp*

- Manage firewall:

systemctl stop iptables

systemctl mask iptables

systemctl start firewalld

systemctl status firewalld

20.3. Configure Firewall for NTP

firewall-cmd --list-all

```
firewall-cmd --permanent --add-service=ntp
firewall-cmd --reload
firewall-cmd --list-all
```

20.4. Configure NTP Server

Edit /etc/ntp.conf:

1. Comment out default pools:

```
#server 0.rhel.pool.ntp.org iburst
#server 1.rhel.pool.ntp.org iburst
#server 2.rhel.pool.ntp.org iburst
#server 3.rhel.pool.ntp.org iburst
```

2. Add your own NTP pool or servers:

```
server 127.127.1.0
fudge 127.127.1.0 stratum 6
```

3. Allow local network access:

```
restrict 192.168.10.0 mask 255.255.255.0 nomodify notrap
```

4. Optional: Create NTP log file:

```
logfile /var/log/ntp.log
```

- Save and exit: :wq
- Start NTP service:

```
systemctl start ntpd
systemctl status ntpd
ntpq -p   # show sources of date/time
ntpq      # check peers
peers
```

20.5. Configure NTP on Linux Client

- Install NTP:

```
yum -y install ntp
```

- Edit /etc/ntp.conf:
  - Comment out default pools.
  - Add local NTP server:

```
server 192.168.10.1
```

- Allow NTP through firewall:

```
firewall-cmd --list-all
firewall-cmd --permanent --add-service=ntp
firewall-cmd --reload
```

- Start and check NTP service:

```
systemctl start ntpd
```

systemctl status ntpd

- Force sync time:

ntpdate -u 192.168.10.1

20.6. Configure NTP on Windows Client

1. Ensure client has IP and can reach server.
2. Right-click **Time** in notification bar → **Adjust date/time** → **Internet Time** → **Change settings**.
3. Enter server IP: 192.168.10.1 → **Update now**.

20.7. Summary

- NTP ensures **synchronized clocks** across servers and clients.
- Proper firewall and service configuration is necessary.
- Both Linux and Windows clients can synchronize with a central NTP server.
- Critical for **logging, security, and network operations**.

Conclusion

This course equips learners with essential Linux administration skills required for managing **modern IT infrastructure**. By mastering services, networking, security, and server configuration, learners can:

- Ensure **secure and efficient system operation**.
- Maintain **network services** and troubleshoot issues effectively.
- Deploy critical servers such as **DNS, DHCP, NTP, Web, Samba, and FTP**.
- Apply best practices for **firewall and service management**.
- Gain the confidence to operate in **enterprise Linux environments**.

Upon completing this course, learners will be prepared to **administer Linux servers professionally**, configure network services, and maintain a secure and optimized IT environment.

References

1. Red Hat, Inc. (2024). *Red Hat Enterprise Linux 7 System Administrator's Guide.* Retrieved from https://access.redhat.com/documentation
2. Nemeth, E., Snyder, G., Hein, T. R., Whaley, B., & Mackin, D. (2017). *UNIX and Linux System Administration Handbook* (5th ed.). Addison-Wesley Professional.
3. Sobell, M. G. (2018). *A Practical Guide to Linux Commands, Editors, and Shell Programming* (4th ed.). Addison-Wesley Professional.
4. Barrett, D., Silverman, R., & Byrnes, R. (2014). *SSH, The Secure Shell: The Definitive Guide.* O'Reilly Media.
5. Shotts, W. (2019). *The Linux Command Line: A Complete Introduction* (2nd ed.). No Starch Press.
6. DNS, DHCP, NTP, and Other Network Services Documentation. (2023). *Linux Documentation Project.* Retrieved from https://tldp.org
7. Nemeth, E. (2018). *Linux Network Administration Guide.* O'Reilly Media.